Jesus Otteson

Midterm 2 Code

1. For our first example, since we are trying to pull the 5 most important pieces of information from our array, the best structure to use to manage these items of information would be Heap Sort. Heap Sort would be able to pull the 5 highest or lowest elements that the fault line sensors would pick up and float them to the top while organizing the array and having the next numbers ready to be pulled out.

Heap Sort also has a big O time of N log(N) consistently making it reliable regardless of the amount of information going through it which would be necessary when updating a website for users to use. They will want to information presented to them at the same time every time. Also, as additional information is presented, when sorting this information if the level of importance is higher for this piece of information it will be floated to the top.

1. Since we are gathering new information that 287 different sensors have been collecting over a month, we are going to have a lot of information to sort through. Since we are looking for something that will run at consistent time I would say we should use Merge Sort. Merge Sort has a consistent time of N log(N) and will be able to take large sums of data and rather than organizing it one element at a time like some sorting algorithms, it will divide the data into different sections and organize many different items of information at the same time.
2. If we are looking for something stable and fast, with the amount of information we are working with we will want to go with Insertion Sort. Insertion sort has a whopping O(1) time and has the stability we are looking for. It will compare our items and organize our new items with our already established items. Since we are combining items rather than organizing unorganized items this will work well without having to worry about the disadvantages of having Insertion Sort organize large sets of unorganized data.